Matriz: Desarrollar la clase "Matriz" que representa una matriz de números en coma flotante y que además permite realizar operaciones sobre ellos.

La clase Matriz tiene los siguientes atributos:

- **int n_filas**: atributo que almacena de forma privada el número de filas de la matriz.
- **int n_columnas**: atributo que almacena de forma privada el número de columnas de la matriz.
- **double **matriz.** Puntero a una matriz bidimensional que permitirá almacenar n_filas x n_columnas elementos de tipo double. Esta matriz deberá ser reservada de forma dinámica en sus constructores y ser liberada en su destructor.

Para esta clase deberán implementarse los siguientes métodos públicos:

- **Constructor por defecto.** Creará una matriz que no tiene elementos por tanto el número de filas y columnas será 0 y el puntero matriz deberá apuntar a NULL.
- **Constructor por parámetros. Matriz(int n_filas, int n_columnas)** Inicializará los atributos n_filas y n_columnas y deberá reservar de forma dinámica espacio suficiente para poder almacenar una matriz con las dimensiones solicitadas.
- **Constructor copia.** Este constructor recibirá como parámetro una matriz y deberá inicializar su matriz con las mismas dimensiones que las indicadas por el parámetro y además se deberán copiar sus elementos.
- **Destructor.** Se encargará de liberar la memoria que fue reservada de forma dinámica para almacenar la matriz.
- **Operador suma y resta.** Se deberá implementar la sobrecarga del operador suma que permite sumar o restar dos matrices que tienen las mismas dimensiones. Este método deberá crear una matriz nueva con las dimensiones de las originales y que contiene la suma o resta de ellas elemento a elemento.
- **Operador producto por escalar.** Se deberá implementar la sobrecarga del operador producto para la multiplicación por un escalar. Este método deberá crear una matriz nueva con las dimensiones de las originales y cuyos elementos son los de la matriz original multiplicados por el escalar.
- **Operador producto por una matriz.** Este método deberá permitir multiplicar dos matrices que deberán tener dimensiones compatibles según la operación de producto. El resultado de esta operación será una matriz con el producto de ambas matrices. NOTA: el producto de dos matrices produce como resultado una matriz con un número de filas igual que el primer operando y un número de columnas igual al segundo operando.
- **Matriz calcularTraspuesta().** Se deberá crear una matriz para almacenar la matriz traspuesta de la original. Una matriz traspuesta es aquella en la que cada elemento (i,j) de la matriz original se encuentra en la posición (j,i) de la matriz destino.

- **bool esSimetrica().** Este método deberá indicar si la matriz es simétrico o no. Se considera que una matriz cuadrad es simétrica si los elementos que guardan una posición simétrica respecto de la diagonal principal son iguales.
- **double obtenerMaximo() y double obtenerMinimo().**Estos métodos deberán devolver el elemento con el máximo o mínimo valor de toda la matriz.

Para facilitar la corrección de los ejercicios mediante el corrector automático se han implementado los siguientes métodos que se pueden encontrar en el fichero Matriz.cpp :

- **void rellenarManual().** Este método permite rellenar una matriz de forma manual solicitando al usuario elemento a elemento.
- **void rellenarAleatorio(long seed).** Este método permite rellenar una matriz de forma aleatoria a partir de la semilla introducida por el usuario.
- **void mostrarMatriz().** Este método permite sacar por pantalla todos los elementos de la matriz.
- **Operador de asignación.** Este método permite asignar una matriz origen a otra matriz destino.

NOTA: Para cada uno de los métodos implementados se deberá incluir una pequeña descripción de su funcionamiento, sus precondiciones mediante assert si las hubiera y el análisis de su complejidad temporal y espacial. Esta información deberá incluirse en la cabecera de cada función.